Machine Translated by Google

CH55X assembly instruction description

Applicable models include: CH551, CH552, CH553, CH554, CH555, CH557, CH558, CH559

Overview of CH55X assembly instruction cycle:

a. The number of instruction cycles of non-jump instructions is the same as the number of instruction bytes (except the DIV instruction of CH551/2/3/4); b. Most instructions are single-byte, 1T single-cycle instructions; c. Jump Instructions containing MOVC/RET/CALL usually have several cycles more than the number of instruction bytes; d.RET/RETI instructions start with 3 cycles, if the return address is an odd number, add 1 cycle; e.MOVC instructions start with 4 cycles, If the address of this instruction is an odd number, add 1 cycle, and the next instruction address is an odd number, plus 1 cycle; f.JMP@A instruction starts from 2 cycles, if the address of this instruction is an odd number, add 1 cycle, if the target address is an odd number Add 1 cycle; g.JB/JNB/JBC/CJNE_A, dir/DJNZ instructions start with 2 more cycles, if the address of this instruction is an odd number, add 1 cycle, such as the target address

Add 1 cycle to an odd number;

h. The rest of the jump instructions start with 2 more cycles. If the target address is an odd number, add 1 cycle; i. If the conditional jump instruction does not jump, the number of cycles is the same as the number of instruction bytes. Note:

The above cycle refers to the reciprocal of the current system main frequency of CH55X.

category	command format	Functional description	command byte	instruction cycle
Transmission instructions (29 items)	MOV A,Rn	(Rn)ÿ(A) The content in Rn is sent to the accumulator A, Rn=R0-R7 (direct)ÿ(A) The content in the	1	1
	MOV A,direct	direct unit address is sent to the accumulator A	2	2
	MOV A,@Ri	(Ri)ÿ(A) The content in the address unit pointed to by the content of Ri is sent to the accumulator A, Ri=R0 or R1 #dataÿ(A) The immediate	1	1
	MOV A,#data	value is sent to the accumulator A	2	2
	MOV Rn,A	(A)ÿ(Rn) The contents of the accumulator A are sent to the register Rn (direct)ÿ(Rn) The	1	1
	MOV Rn,direct	contents of the direct addressing unit are sent to the register #dataÿ(Rn) The immediate value is	2	2
	MOV Rn,#data	directly sent to the register Rn	2	2
	MOV direct,A	(A)ÿ(direct) The content in the accumulator A is sent to the direct addressing unit 2		2
	MOV direct,Rn	The content in (Rn)ÿ(direct) register is sent to the direct addressing unit (direct1)ÿ(direct2) The	2	2
	MOV direct2,direct1	content in the direct addressing unit 1 is sent to the direct addressing unit 2	3	3
	MOV direct,@Ri	(Rn)ÿ(direct)The content in the register is sent to the direct addressing unit	2	2
	MOV direct, #data #dataÿ(direct) send immediate data to direct addressing unit		3	3
	MOV @Ri,A	(A)ÿ((Ri)) The content in the accumulator A is sent to the RAM unit whose address is the content in Ri	1	1
	MOV @Ri,direct	(direct)ÿ((Ri)) Send the content of the direct addressing unit to the RAM unit whose address is the content in Ri #dataÿ((Ri)) Immediate data is sent	2	2
	MOV @Ri,#data	to the RAM unit whose address is the content in Ri RAM unit	2	2
	MOV DPTR,#data16	#dataHÿ(DPH), #dataLÿ(DPL) The upper 8 bits of the 16-bit constant are sent to DPH, and the lower 8 bits are sent to DPL	3	3
	MOVX A,@DPTR	((DPTR))ÿ(A) The content of the data pointer pointing to the external RAM (16-bit address) is sent to the accumulator A	1	1
	MOVX A,@Ri	((Ri))ÿ(A) The register Ri points to the content of the external RAM (8-bit address) and sends it to the accumulator A, and the high-order 8-bit address is provided by P2	1	1
	MOVX @DPTR,A	(A)ÿ((DPTR)) The content in the accumulator is sent to the data pointer pointing out 1		1

Machine Translated by Google

		Internal RAM (16-bit address)		
	MOVX @Ri,A	(A)ÿ((Ri)) The content in the accumulator is sent to the register Ri to point to the outside RAM (8-bit address), the upper 8-bit address is provided by P2	1	1
	MOVC A,@A+DPTR CHILDREN	((A)+(DPTR))ÿ(A) The content in the table address unit is sent to the accumulator A	1	5/6/7
	MOVC A,@A+PC	(PC)+1ÿ(PC), ((A)+(PC))ÿ(A) The content in the table address unit is sent to the accumulator A	1	5/6/7
	XCH A,Rn	(A)ÿÿ(Rn) the accumulator and the working register Rn 1 (direct)ÿ(A) The content of the accumulator A is exchanged with the content of		1
	XCH A,direct	the direct addressing unit (A)ÿÿ((Ri)) The content of the accumulator and the storage unit pointed	2	2
	XCH A,@Ri	to by the working register Ri are exchanged Swap (A[3:0])ÿÿ((Ri)[3:0]) the accumulator and the contents of the storage	1	1
	XCHD A,@Ri	unit pointed to by the working register Ri swap (A[3:0])ÿ ÿ(A[7:4]) The high and low nibbles in the accumulator are swapped (SP)+1ÿ(SP), (direct)ÿ(SP) The	1	1
	SWAP A	stack pointer is first increased by 1, and the data in the direct addressing unit Send to the unit	1	1
	PUSH direct	pointed to by the stack pointer SP (SP)ÿ(direct), (SP)-1ÿ(SP) The unit data pointed to by the stack pointer SP is sent to the direct addressing unit, and the stack pointer SP is decremented	2	2
	POP direct	by 1 Operation (A)+(Rn)ÿ(A) The content in the accumulator A is added to the content in the working register Rn, and the result is stored in A (A)+(direct)ÿ(A) The content in the accumulator A and The	2	2
Arithmetic operation instructions (24)	ADD A,Rn	contents of the direct address units are added, and the result is stored in A.	1	1
	ADD A,direct		2	2
	ADD A,@Ri	(A)+((Ri))ÿ(A) The content in the accumulator A is added to the content in the address unit pointed to by the working register Ri, and the result is stored in A	1	1
	ADD A,#data	(A)+#dataÿ(A) The content in the accumulator A is added to the immediate value #data, and the result is stored in A	2	2
	ADDC A,Rn	(A)+Rn+(C)ÿ(A) The content in the accumulator A is added to the content in the working register Rn, together with the carry bit, and the result is stored in A	1	1
	ADDC A, direct	(A)+(direct)+(C)ÿ(A) The content in the accumulator A is added to the content of the direct address unit together with the carry bit, and the result is stored in A	2	2
	ADDC A,@Ri	(A)+((Ri))+(C)ÿ(A) The content in the accumulator A and the content in the address unit pointed to by the working register Ri are added together with the carry bit, and the result is stored in A	1	1
	ADDC A, #data	(A)+#data+(C)ÿ(A) The content in the accumulator A is added to the immediate value and the carry bit, and the result is stored in A	2	2
	INC A	(A)+1ÿ(A) Add 1 to the content of accumulator A, and store the result in A	1	1
	INC Rn	(Rn)+1ÿ(Rn) Add 1 to the content of register Rn, and send the result back to the original address	1	1
	INC direct	unit (direct)+1ÿ(direct) Add 1 to the content in the direct address unit, and send the result back to the original address unit	2	2
	INC @Ri	((Ri))+1ÿ((Ri)) Add 1 to the content of the address unit pointed to by the content of the register, and send the result back to the original address unit	1	1

Machine Translated by Google

	INC DPTR	(DPTR)+1ÿ(DPTR) The content of the data pointer is increased by 1, and the result is sent back to the data pointer	1	1
	SUBB A,Rn	(A)-(Rn)-(C)ÿ(A) The content in accumulator A is subtracted from the content in the working register, together with the borrow bit, and the result is stored in A	1	1
	SUBB A,direct	(A)-(direct)-(C)ÿ(A) The content in accumulator A is subtracted from the content in the direct address unit, together with the borrow bit, and the result is stored in A	2	2
	SUBB A,@Ri	(A)-((Ri))-(C)ÿ(A) The content in accumulator A is subtracted from the content in the address unit pointed to by working register Ri, together with the borrow bit, and the result is stored in A	1	1
	SUBB A,#data	(A)-#data-(C)ÿ(A) The content in accumulator A is subtracted from the immediate value and the borrow bit, and the result is stored in A	2	2
	DEC A	(A)-1ÿ(A) The content in the accumulator A is subtracted by 1, and the result is sent back to the accumulator A	1	1
	DEC Rn	(Rn)-1ÿ(Rn) Subtract 1 from the content in register Rn, and send the result back to register Rn	1	1
	DEC direct	(direct)-1ÿ(direct) The content in the direct address unit is decremented by 1, and the result is sent back to the direct address unit	2	2
	DEC @Ri	((Ri))-1ÿ((Ri)) The content in the address unit pointed to by the register Ri is decremented by 1, and the result is sent back to the original address unit	1	1
	MUL AB	(A)×(B)ÿ(A) and (B) Accumulator A The content in register B is multiplied by the content in register B, and the result is stored in A and B (A)÷(B)ÿ(A) and (B)	1	1
	DIV AB (CH555/557/558/559)	The content in accumulator A is divided by the content in register B, and the obtained The quotient is stored in the accumulator A, and the remainder is stored in the register B (A)÷(B)ÿ(A) and (B) The content in	1	1
	DIV AB (CH551/552/553/554)	the accumulator A is divided by the content in the register B, and the resulting quotient is stored in the accumulator A, The remainder is stored in the accumulator in register B, and performs decimal conversion.	1	4
	And A	The content of accumulator A and the	1	1
Logical operation instructions (24)	ANL A,Rn	content in register Rn perform AND logic operation, and the result is stored in accumulator A. The content in accumulator A and the content in	1	1
	ANL A,direct	the direct address unit perform AND Logical operation, the result is stored in the accumulator A in the register A and the content in the address unit	2	2
	ANL A,@Ri	pointed to by the working register Ri performs AND logical operation, the result is stored in the accumulator A and the immediate value Perform AND logical operation. The	1	1
	ANL A,#date	result is stored in the direct address unit in the accumulator A and the content of the accumulator A is performed AND	2	2
	ANL direct,A	logic operation, the result is stored in the direct address unit in the direct address unit and the immediate value is performed and logical operation, the	2	2
	ANL direct,#data	result is accumulated in the direct address unit The contents of accumulator A and the contents of register Rn are logically ORed, and the	3	3
	ORL A,Rn	result is stored in accumulator A. 1 in address location pointed to by working register Ri	1	1
	ORL A,direct		2	2
	ORL A,@Ri			1

Machine Translated by Google

		OR the contents, and the result is stored in accumulator A The contents		
	ORL A,#data	of accumulator A and the immediate value are ORed, and the result is stored in the direct address	2	2
	ORL direct,A	location of accumulator A, and the result is stored in accumulator A The content in the direct address unit in the direct address unit and	2	2
	ORL direct,#data	the immediate value perform a logic OR operation, and the result is stored in the accumulator A in the direct address	3	3
	XRL A,Rn	unit and the content in the register Rn performs a logic exclusive OR operation, and the result is stored in the accumulator A	1	1
	XRL A,direct	Execute logical XOR operation between the content in the direct address unit and the content in the direct address unit, the result is	2	2
	XRL A,@Ri	stored in the accumulator A in register A The content in the address unit pointed to by the working register Ri performs a logical exclusive OR operation, and the	1	1
	XRL A,#data	result is stored in the accumulator A Execute logical XOR operation between the content of A and the	2	2
	XRL direct,A	immediate value, and the result is stored in the content of the direct address unit in accumulator A and the content of accumulator A, and	2	2
	XRL direct,#data	the result is stored in the content of the direct address unit in the direct address unit and Execute the logical XOR operation	3	3
	CLR A	on the immediate value, and the result is stored	1	1
	CPL A	in the direct address unit 0ÿ(A), and the	1	1
	RL A	content in the accumulator is cleared to 0.	1	1
	RR A	The content in the accumulator is bit-	1	1
	RLC A	inverted, and the content in the accumulator A is shifted left by one	1	1
	RRC A	bit in the accumulator A. The contents of accumulator A are shifted	1	1
Control Transfer Instructions (16)	LJMP addr16	right by one bit together with the carry bit CY. The contents of accumulator A are shifted right by one bit together with the carry bit CY	3	5/6
	AJMP addr11	(PC)+2ÿ(PC), addr11ÿ(PC[10:0]) the program counter is given a new value (11-bit address), (PC[15:11]) does not change.	2	4/5
	SJMP rel	(PC)+2+relÿ(PC) Add 2 to the current program counter and then add the offset to assign a new value to the program counter.	2	4/5
	JMP @A+DPTR CHILDREN	(A)+(DPTR)ÿ(PC), the value of the address unit pointed to by the accumulator plus the value of the data pointer gives the program counter a new value.	1	3/4/5
	JZ rel	A=0, (PC)+2+relÿ(PC), the content in the accumulator is 0, then transfer to the address pointed by the offset, otherwise the program will execute Aÿ0, (PC)+2+	2	2 or 4/5
	JNZ rel	relÿ(PC), the content in the accumulator is not 0, then transfer to the address pointed by the offset, otherwise the program will execute Aÿ#data, (PC)+3+relÿ(PC),	2	2 or 4/5
	CJNE A,#data,rel	the accumulator If the content in the working register is not equal to the immediate value, it will be transferred to the address pointed by the offset, otherwise the program will execute	3	3 or 5/6
	CJNE Rn,#data,rel	Aÿ#data, (PC)+3+relÿ(PC), and the content in the working register Rn is not equal to the immediate number, then transfer to the address pointed by the offset, otherwise the program will execute	3	3 or 5/6
	CJNE @Ri,#data,rel	Aÿ#data, (PC)+3+relÿ(PC), and the content in the address unit pointed to by the working register Ri is not equal to the immediate value, then Transfer to the address pointed by the offset, otherwise the program will continue to execute	3	3 or 5/6

Machine Translated by Google

	CJNE A,direct,rel	Aÿ(direct), (PC)+3+relÿ(PC), if the content in the accumulator is not equal to the content of the direct address unit, transfer to the address pointed by the offset, otherwise the program will continue to execute	3	3 or 5/6/7
	DJNZ Rn,rel	(Rn)-1ÿ(Rn), (Rn)ÿ0, (PC)+2+relÿ(PC) If the working register Rn minus 1 is not equal to 0, then transfer to the address pointed by the offset, otherwise the program goes to Next execute (direct)-1ÿ(direct),	2	2 or 4/5/6
	DJNZ direct,rel	(direct)ÿ0, (PC)+2+relÿ (PC) the content in the direct address unit minus 1 is not equal to 0, then transfer to the address pointed by the offset address, otherwise the program executes the long call instruction downwards, and the subroutine	3	3 or 5/6/7
	LCALL addr16	can be called in the 64KB space. At this time (PC)+3ÿ(PC), (SP)+1ÿ(SP), (PC[7:0])ÿ(SP), (SP) +1ÿ(SP), (PC[15: 8])ÿ(SP), addr16ÿ(PC), that is, the return address pushed in when the subroutine is called is popped from the stack respectively. Absolute call instruction, can call subroutine in 2KB space, at this time (PC)+2ÿ(PC), (SP)+1ÿ(SP),	3	5/6
	ACALL addr11	(PC[7:0])ÿ(SP), (SP) +1ÿ (SP), (PC[15:8])ÿ(SP), addr11ÿ(PC[10:0])	2	4/5
	RIGHT	Subroutine return	1	4/5
	RARELY	instruction Interrupt return instruction, RETI instruction cannot use	1	4/5
Bit manipulation instructions (18)	MOV C,bit	RET instead of bitÿCY, a certain bit of data is sent to CY	2	2
	MOV bit,C	CYÿbit, CY data send a certain bit 0ÿCY,	2	2
	CLR C	clear CY 0ÿbit, clear	1	1
	CLR bit	a certain bit 1ÿCY, set CY	2	2
	OLD SETB	1ÿbit, set a certain bit	1	1
	SETB bit		2	2
	CPL C	CY Invert bit	1	1
	CPL bit	Invert	2	2
	ANL C,bit	CY and bit are ANDed, and the result is stored in CY	2	2
	ANL C,/bit	Inverse AND of CY and bit, the result is stored in CY	2	2
	ORL C,bit	CY and bit phase OR, the result is stored in CY	2	2
	ORL C,/bit	Inverse OR of CY and bit, the result is stored in CY	2	2
	JC rel	(CY)=1 transfer, (PC)+2+relÿPC, otherwise the program will execute down, (PC)+2ÿPC	2	2 or 4/5
	JNC rel	(CY)=0 transfer, (PC)+2+relÿPC, otherwise the program will go down, (PC)+2ÿPC bit	2	2 or 4/5
	JB bit,rel	state is 1 transfer, (PC)+3+relÿPC, otherwise the program continues to execute, (PC)+3ÿPC bit state is 0 transfer, (PC)	3	3 or 5/6/7
	JNB bit,rel	+3+relÿ PC, otherwise the program is executed downwards, (PC)+3ÿPC bit state is 1, and the bit is cleared to 0, (PC)	3	3 or 5/6/7
	JBC bit,rel	+3+relÿPC, otherwise the program is executed downwards, (PC)+3ÿ PC empty instruction, this instruction does not perform any operation	3	3 or 5/6/7
	NOP	except to increase PC by 1 and consume one cycle, which can be used for a short time delay	1	1
new command	DB 0xA5	XRAM fast copy instruction	1	1

Note: The mark is gray, indicating that the number of bytes of the instruction is not equal to the cycle required to execute the instruction

Machine Translated by Google

Example of using the new command:

```
;New Instruction: MOVX @DPTR1,A
;Instruction Code: 0xA5
;Instruction Cycle: 1
;Instruction Operation:  ;step-1.
    write ACC into xdata SRAM @DPTR1 embedded chip
    ;step-2. increase DPTR1
ASM example:
        INC XBUS_AUX
        MOV DPTR,#TARGET_ADDR ;DPTR1
        DEC XBUS_AUX
        MOV DPTR,#SOURCE_ADDR ;DPTR0
        MOV R7,#xxH
LOOP: MOVX A,@DPTR              ;DPTR0
        INC DPTR                ;DPTR0, if need
        DB 0A5H                 ;MOVX @DPTR1,A & INC DPTR1
        DJNZ R7, LOOP
```